Contract DAAB07-98-C-B601

Automated Tool Suite (ATS)
SBIR Phase 1
Final Report

May 29, 1998

Submitted To:

CECOM, C2SID

Submitted By:

PREDICTION SYSTEMS, INC.
309 Morris Avenue
Spring Lake, NJ 07762

(732)449-6800 (732)449-0897
psi@predictsys.com www.predictsys.com

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
System Architecture Tool Design

5. FUNDING NUMBERS
DAAB07-98-C-B601

6. AUTHOR(S)
Jose L. Ucles, Larry Dworkin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Prediction Systems, Inc.
309 Morris Avenue.
Suite G
Spring Lake, New Jersey 07762

8. PERFORMING ORGANIZATION REPORT NUMBER
PSI-97001

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Commander US Army CECOM
AMSEL-AGCA-B-AL
Fort Monmouth, New Jersey 07703

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
None

12a. DISTRIBUTION/AVAILABILITY STATEMENT
UNLIMITED

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*) - Report Developed under SBIR contract

In order to achieve a well-conceived system architecture several steps are involved. Initially, system requirements need to be defined, functional procedures need to be established, an operational architecture needs to be developed, and a System Architecture needs to be realized. The PSI/MU Team proposes to develop a specification and a concept of operation for the Automated Tool Suite (ATS) which will take the various requirements of the Army's Operational Architecture and automate the synthesis of the system architecture. What makes this Automated Tool Suite (ATS) unique is its focus on offering the system architect designer a series of steps that guide the user through the complex process of developing the architecture. They include: (1) a user-friendly interface that includes an expert system which allows various architectural tradeoffs, (2) a series of supporting databases that help in rapidly defining system requirements in support of the operational requirements, (3) an expert system that interacts with high level parametric models that are used to provide the System Architect feedback about expected performance of proposed architectures, and (4) an analyst's back end that permits the viewing of the architecture from several views (e.g., hardware, communication and network, etc.) useful to an architect.

14. SUBJECT TERMS SBIR Report
Architecture Tool Suite, System Architecture, User Requirements, Architecture News

15. NUMBER OF PAGES
53

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

Table of Contents

1 Introduction

The overall objective of this Phase 1 effort was to develop a specification and concept of operation for an Automated Tool Suite (ATS). The ATS will assist a user in taking various representations of an Operational Architecture and will help translate these into a System Architecture. The ATS will offer a user the following:

- A user-friendly interface which allows architectural trade offs, a series of supporting databases taken from operational architecture representation and a knowledge-based system capable of assisting in developing a system architecture.
- An expert system that interacts with high-level parametric models that is used to provide the system architect feedback about expected performance of a proposed architecture.
- An analysts' back end that permits the viewing of simulation results from several perspectives.

This product has potential for use both in the military and commercial sector. A wide variety of simulation and modeling products exist on the market (e.g., GSS, OPNET, and COMNET), but it does not appear that either of these products has been integrated into an architecture tool. PSI has had the experience in supporting the development of the Architecture Assessment Tool (AAT), a tool that was used to evaluate at a high level the architecture that the NATO countries should pursue for the Year 2015. Several architectural tradeoffs were made to decide in the proper protocols to use for that architecture. However, this analysis was geared towards the development of a Technical Architecture, rather than a System Architecture.

To automate the process as much as possible, the ATS will also allow the creation of input files by interactions with commercial database packages that define user requirements (i.e., an Operational Architecture) automatically. Additionally the user will be able to see the results of the proposed architecture graphically. The ATS can complement these products and permit the more effective use of these simulation tools. It is proposed that this Phase II will take the results of the Phase I effort and conduct the necessary development to yield a well-defined deliverable product suitable for commercialization and export to potential user groups.

1.2 Phase I Accomplishments

The Phase I consisted of the following deliverables: (1) a Domain Analysis Report, (2) a Concept of Operation, (3) an Initial Specification Report, (4) an Architecture Design Report, and (5) a Software Specification Report for the ATS prototype along with the corresponding developed software. A summary of the most relevant deliverables is included in the following sections.

2 Domain Analysis

This effort was accomplished by collecting, organizing, and representing relevant information in the domain of Information Transfer Architectural Development. This section is presented in three parts that include the following: data collection, context analysis and domain modeling.

Data Collection

A review of relevant literature was conducted. Of particular interest was the Army Enterprise Architecture (AEA), which defines the DOD's C^4ISR Architecture Framework. This document defines the management process for developing and maintaining a comprehensive, integrated information blueprint that translates Army operational patterns into system architectures that support warfighting systems. Below is a matrix that lists the key products needed to complete systems architecture. (The tools described for the development of a system architecture are very limited.) These products are needed, but no mention is made of how automation can assist in the creation of OA products or its evolution into a system architecture.

References 3-8 deal with various automated tools that assist in the creation of an OA and its evaluation into a system architecture. These tools included:

1. Netviz – A graphical/database tool for storing and viewing elements of an operational architecture.
2. Live Analyst – A tool to test critical operational attributes/architectures such as time, cost, and capacity, etc., so a process can be realistically analyzed. When applied to the military, it can be used to assess how to support command and control operations
3. Architecture Attribute Analysis Aid – A4 –A tool developed at USC as an aid to developing systems architecture from multiple perspectives which include: cost, schedule, reliability, attributes, relationships, etc. The tool also has a high-level programming language for conversion of specifications into object oriented code.
4. Performance Modeling Tools – OPNET, GSS, AMASE, TIMS Models that allow a user to examine computer and communication architectures from a performance perspective in order to evaluate alternative systems designs.
5. Economic Analysis Tools – Tools for estimating hardware and software costs based on high-level description of systems design (e.g., COCOMO II).

Context Analysis

The ATS is an automation tool that is software intensive. The products of the context analysis place the architectural development domain within the overall context of Research, Development and Acquisition (RDA) of IT-C^4ISR systems. In order to portray the relationship between RDA and those applications addressed by the ATS, we will use a hierarchy of layers as seen in Figure 2.a. The figure is in the form of a structure diagram. The four primary layers of the ATS are: the hardware layer, the Commercial-Off-the Shelf (COTS) layer, the Common Application Support software layer, and the ATS application software layer. Layers 1 and 2 of the figure show the Common Hardware and Software suites, which are needed to provide an operating environment for the ATS tool. In Layer 1 the hardware can involve a single host or a heterogeneous set of hosts. In Layer 2, the COTS layer, software that has already been built and tested can be tailored to provide software services that support ATS applications. The line between Layer 2 and Layer 3 symbolizes the interface between packages such as UNIX OS and X Windows and supporting applications software.

Layer 3 shows the specific application support software and tools which provide common services that include: performance models, an operational architecture database, and knowledge based tools. Layer 4, the Application Layer, consists of the specific applications the PSI/MU Team envisions for the ATS. These applications must undergo additional examination by CECOM and our contracting team to assure they support the eventual use of ATS in simplifying the system architecture development process.

Each application area is discussed briefly below:

- Economic Analysis – Provides automated tool for the cost of hardware and software when a design is in the preliminary stage.
- Systems Architecture Performance Analysis – Systems modeling tool that provides simulation and analysis environment for evaluating performance of alternative architectures.
- Model Selection -- Rule base tool that helps user select the best performance model for a given application.
- Message Traffic Analysis Tool – Helps develop a traffic model for driving performance and analytical models.
- Exercise Planning and Support Tool – Assists exercise planners with developing IP address book, test plans and data recording.
- Scenario Development – Use of a constructive HLA-compatible model that can provide stimulus to an exercise or performance model.
- Architecture Decision Analysis – Helps user examine several criteria (e.g., cost, survivability, etc.) in order to s elect most cost-effective design.

- Model Initialization – Knowledge-based tool that can assist in the rapid set up and running of performance models based on real unit deployments.
- Advanced Technical Demonstration Planning – Helps lay out projects, cost and alternative design useful in planning future ATD efforts.

ATS Application Software Layer 4	• Economic analysis • Systems architecture Performance analysis • Message traffic analysis • Exercise planning & support • Scenario development • Architecture decision analysis	• Model Initialization • ATD Planning
Common Application Support Software Layer 3	• Netviz representation of O.A.DB • Live Analyst process oriented Tool • Performance models – OPNET, AMASE, GSS • Knowledge based tool – Key, Acquire • Automated tool suite unique Modules	
System Support Software (COTS) Layer 2	• Java • HTML • Web Services	• UNIX OS • X Windows • Visual C++ • Graphics support
Hardware Layer 1	• SunSpark • SGI • H.P.	• P.C.

Figure 2.a Structure Diagram for ATS

Domain Modeling

We developed a feature analysis, entity relationship diagram and a functional analysis. The features analysis was performed by conducting meetings between the ATS developers (PSI/MU) and the potential users (RDEC, PMs). The PSI/MU team has also prepared a *concept of operation*. A full features analysis was completed after RDEC feedback is received.

The relative usefulness of these entities is evaluated in the features analysis shown in Figure 2.b. A three-dimensional graphic is shown listing the potential applications along the **Z** axis, the reusable tools along the **X** axis, and the Features along the **Y** axis. Each tool is scored as to its relative usefulness (shown at the top of the plane) in achieving the applications described earlier in this section. Netviz and Live Analyst appear to have the largest positive impact on achieving an ATS capable of supporting this set of applications. On the right hand portion of the **Z-X** plane, an assessment of how a given application is supported by the set of tools is indicated. A low score indicates that the tool set does not support the given application. As an example, "Model Selection" is not well supported and should involve a new development imbedded in the ATS tool. However, it should be noted that the Government's direction has been to use the OPNET environment as the environment of choice for future model development. This directive takes into account circumstances not included in this domain analysis. As a result, our proposal has OPNET as the environment for model development.

In addition to the usefulness of a given tool, we are also concerned about any special features that may be of consequence. A high score indicates that a given tool has good features of concern to the user.

Thus we can conclude that Netviz and Live Analyst have features that should qualify them for purchase by CECOM/PSI/MU, while applications involving model selection, Architecture Decision Analysis and automation of force laydown will not gain any value from software reuse and should be addressed in the ATS.

Automated Analyst

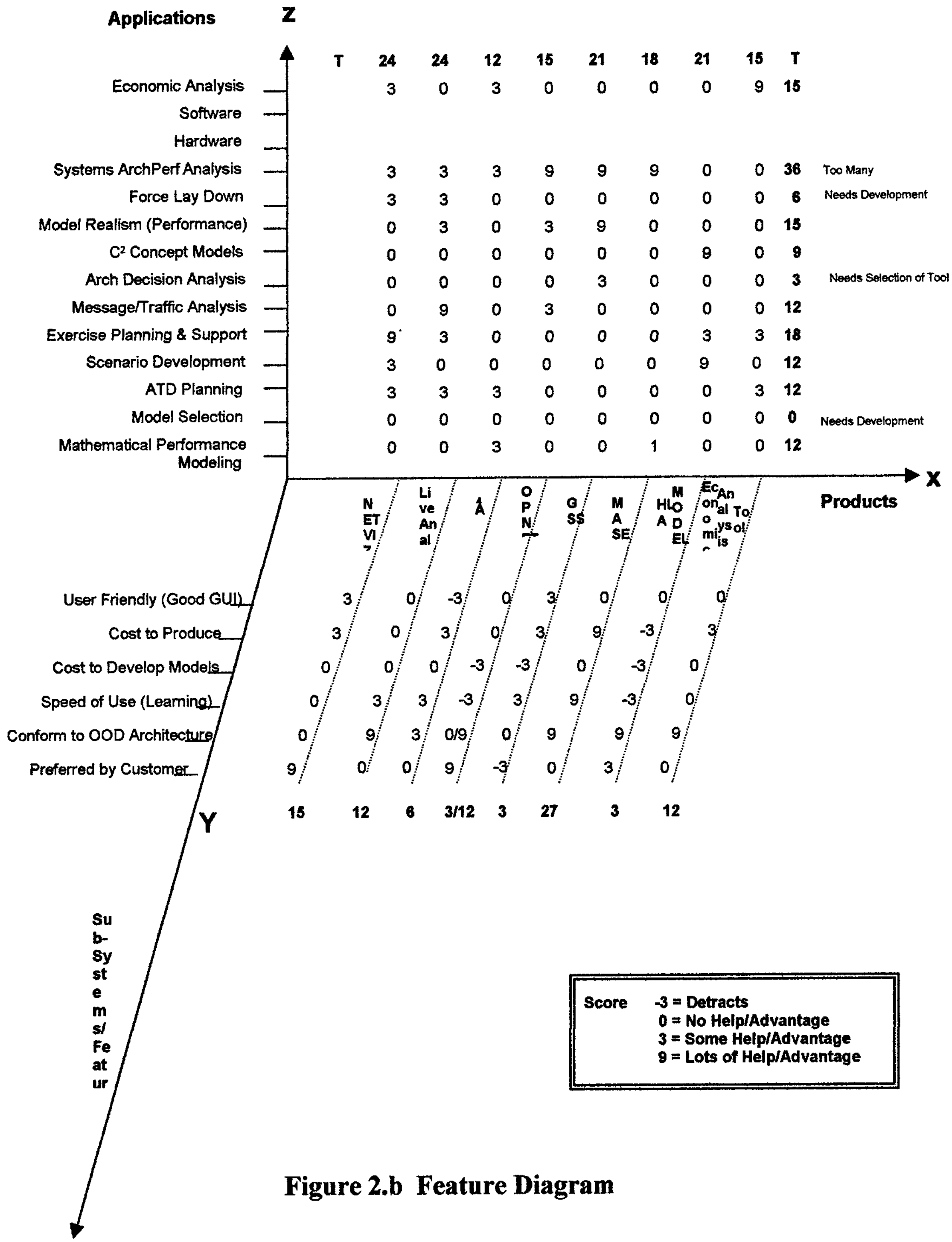


Figure 2.b Feature Diagram

3 Concept of Operation

Figure 3.a illustrates the enhanced development process. Building the database, its architecture, and being able to access the database in all phases of the development process will be one of the major objectives of the ATS development process. Also shown in the figure are the tools and processes needed to move from one step to the next. The underlined elements are those enhancements that were being considered for incorporation into the ATS. For a detailed description of the items that are proposed to be developed under a Phase II effort, refer to the Phase II Work Plan Section.

The purpose of the enhancements is to improve the development process of a system architecture from several perspectives. These include:

- Early elimination of poorer alternative architectures.
- Speed up the process of developing systems architectures compared to current design procedure.
- Lead the designer through a proven design process as shown in Figure 3.a.
- Allow reuse of relevant databases at several steps in the design process saving cost and time.
- Offering the designer a set of inexpensive tools linked in a convenient form that can enhance many of the steps in the design process.
- Using an ATS development environment (e.g., Microsoft Visual Studio) that permits down-steam augmentation by future users of the tool.
- Provide continuous documentation and linkage of all steps in the design process.

The ATS will consist of a set of related screens, tools and databases that lead the user, in a logical fashion through the process of creating a systems architecture from an operational architecture. A step-by-step description of this process will amplify the concept of operation.

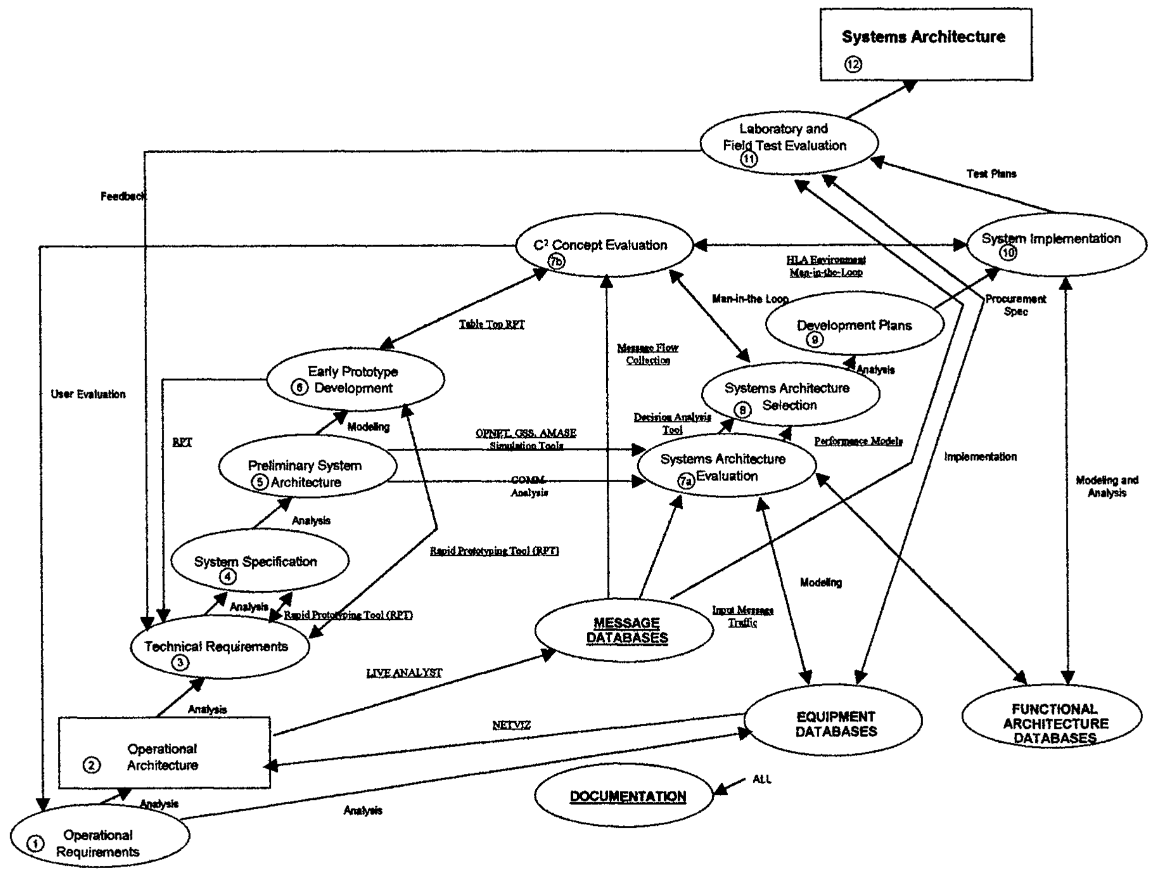


Figure 3.a Enhanced Development Process

When the ATS is turned on, a title page appears as shown below.

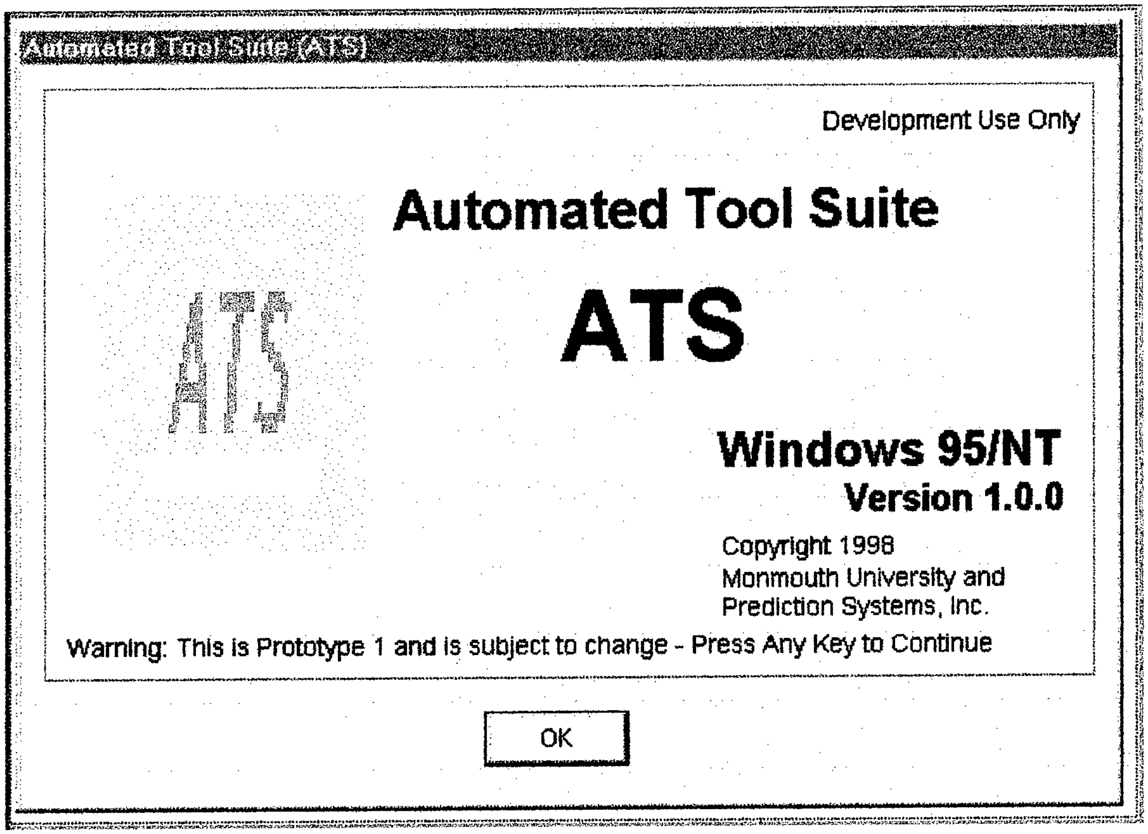


Figure 3.b Title Screen

Upon clicking on the OK box, the screen shown in Figure 3.c appears. The user has the option of starting a new architecture development or embellishing an existing systems architecture. In either case, he/she advances to the screen shown in Figure 3.d.

Shown are the 13 steps and supporting databases along with necessary control buttons (e.g., Exit, Help).

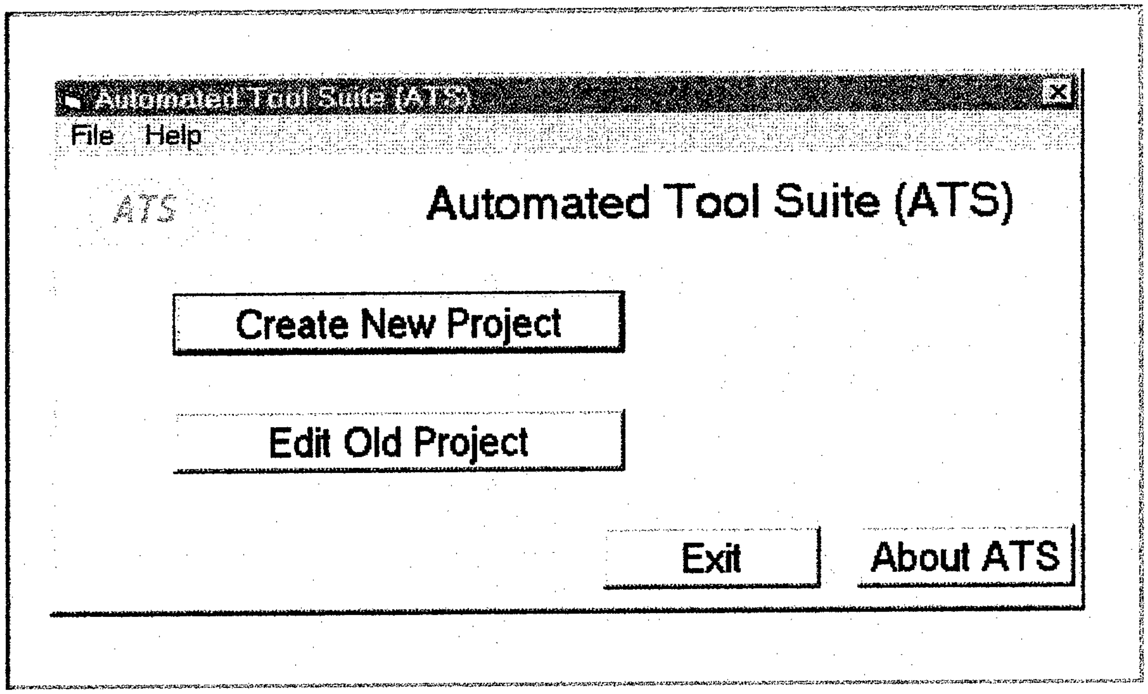


Figure 3.c Architecture Development Options Screen

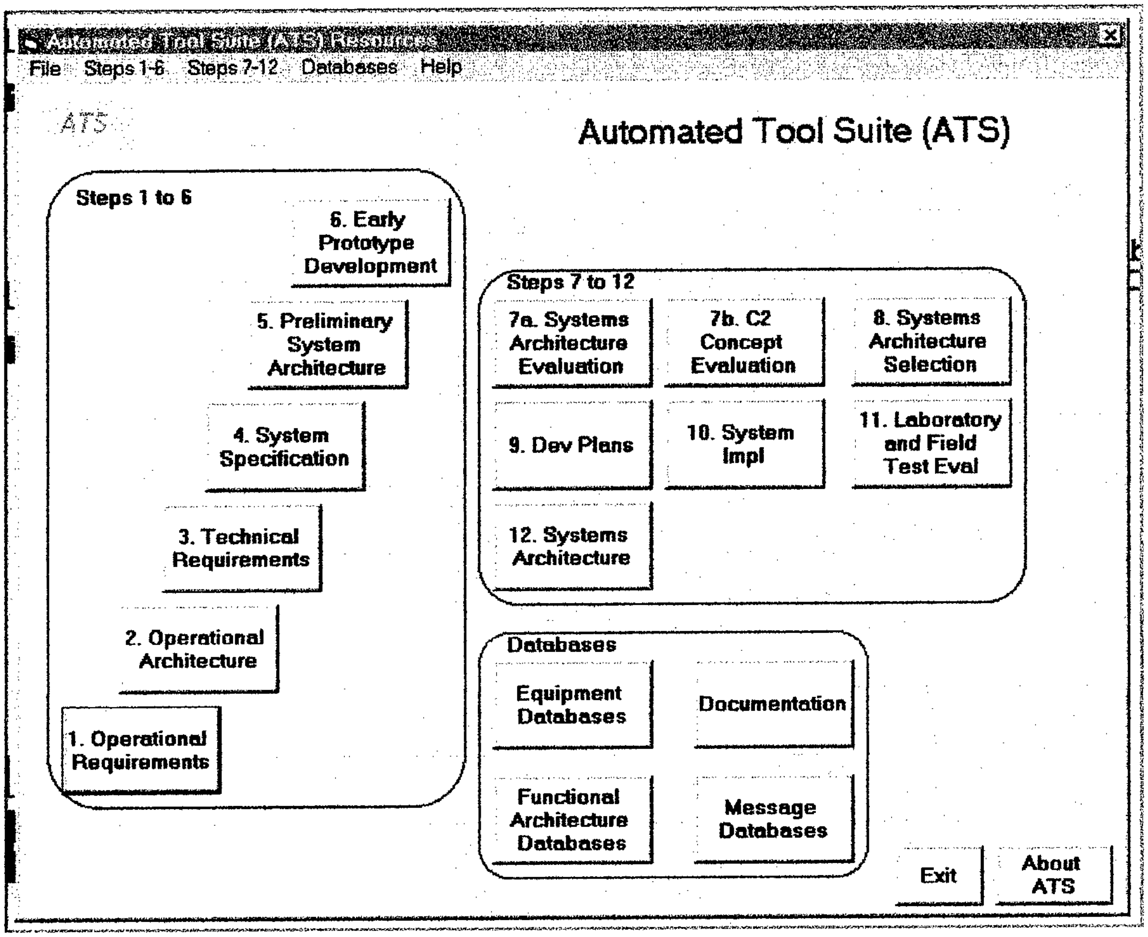


Figure 3.d ATS Process Screen

This screen (Figure 3.d) is based on the process model shown in Figure 3.a. The user can move to any step in the process through the use of the tab key on his keyboard. The user can click on any button and open a "primary control screen" supporting each step shown on the process screen. The primary control screen contains a description of tools and databases linked to that step in the process and describes the principal functions in that step. It also describes how to employ the databases and tools essential to move to other steps in the process. We will describe some of the "primary control screens," the tools and databases available at each step in the process.

The operational architecture step is supported currently by a series of views of the operational architecture. Some of these are contained in the NETVIZ and Live Analyst databases in use by the PEO C3S and FIO. The automated portions of these tools are supported by contractor personnel from Booz Allen & Hamilton, MITRE, Computer Sciences Corporation and Intellicorp. These tools will be used to create supporting databases shown in the process model of Figure 3.a. They will be used to help construct the message database, equipment database, functional architecture database and documentation database. Live Analyst can also serve as a rapid prototyping tool essential in defining technical requirements (Step 3), system specification (Step 4), as well as in creating the preliminary system architecture (Step 5), and the early prototype (Step 6). Linkage to these tools will be achieved by clicking on buttons located on the primary control screen.

If the user chooses to examine his documentation database and clicks either on that button in the Operational Architecture Primary Control screen or directly on the documentation button on the screen shown in Figure 3.d, the documentation screen shown in Figure 3.e will appear. The items listed represent standard software and firmware documents that will be developed for a given system in the process of developing the system's architecture. In the initial prototype ATS, skeletal documents can be found for each item listed in Figure 3.e as indicated by a small white dot in front of it. We will briefly describe other tools available at each step in the process

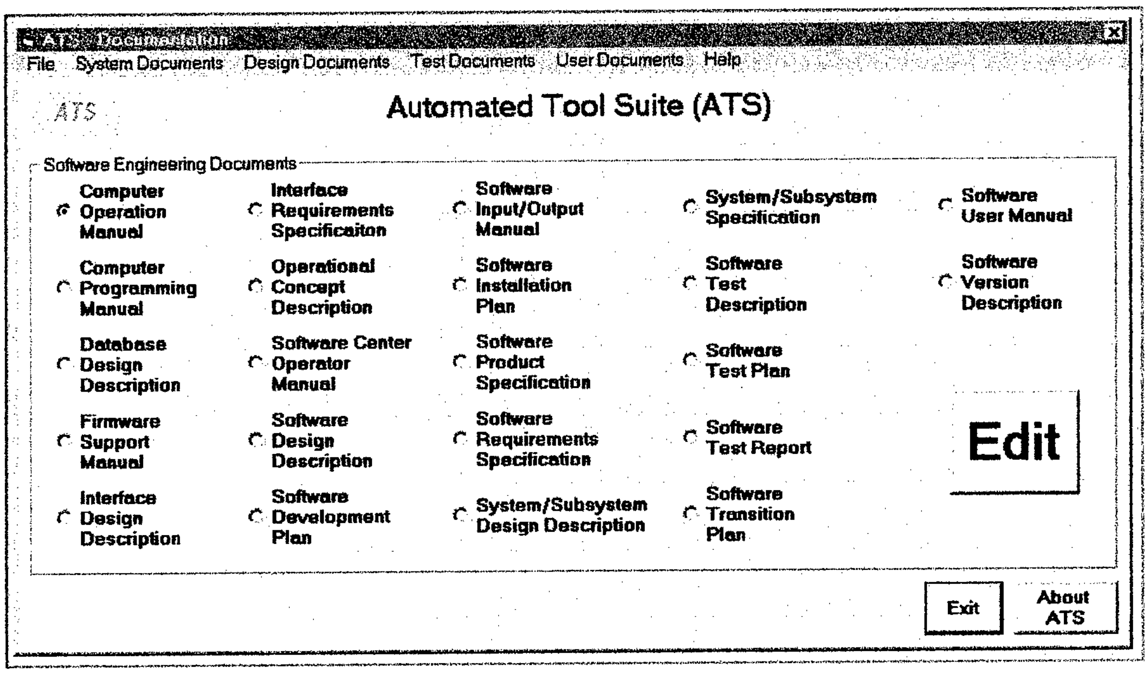


Figure 3.e Documentation Screen

In order to enhance the building of technical requirements and system specifications, templates for creating these documents will be provided via the documents shown in Figure 3.e. In addition, many other tools can be linked to the ATS. These tools can assist in the early representation of a system architecture and are useful in assessing how well preliminary architectures meet technical requirements (e.g., access, excel, etc.). If a system can be examined early in the development processes, it can permit the elimination of poor proposed architectures, saving time and costs. For more detailed examination of preliminary architectures, the "primary control screen" of Step 5 (Preliminary System Architectures) contains a series of buttons linked to available performance evaluation tools. A knowledge-based tool (accessed via push button) can be employed to help select the best performance analysis tool. The user will be asked a series of questions including: the character of the information desired, the time limits associated with the analysis, as well as his/her preference in tool selection. This will lead to designation of a preferred modeling tool. Model

environments include GSS, OPNET and AMASE tools or others as needed. However, as stated previously, OPNET has been selected by the Government as the M&S environment, which will be the one supported for Phase II development. Accessing these tools leads directly into Step 7a, Systems Architecture Evaluation.

One of the more time-consuming efforts has been the building of performance models of new and existing communication architectures in order to evaluate their performance. Step 7a will be supported by three primary databases. These include the message database, equipment database and functional architecture database. Effective use of these databases will significantly shorten the time to create and evaluate a given architecture. The performance model will access the needed databases built during the creation of the operational architecture, the evaluation of legacy systems, and those derived from other analysis efforts. An example of this type of information is the message spreadsheet contained in the previously submitted system specification.

Upon clicking on the Systems Architecture evaluation button, a verbal description of how to use this step of the process appears. Push buttons linking to a particular performance model or one of the ATS databases will be available. Upon completion of the evaluation, results will be stored in the appropriate database for future use or for preparation of needed documents.

In addition, a button in Step 7a will lead to a decision analysis tool. This tool aids in building a decision tree needed to select preferred architectural alternatives.

If one is concerned with development of implementation plans, the designer can progress to Step 9 (Development Plans). The Microsoft family of tools offers a set of tools to assist in this process including Microsoft Project, that is readily accessible to the Microsoft Visual Studio Tool Suite, or Visual J++ which is being proposed for the Phase II effort.

In this fashion the user of the ATS is lead through a systematic procedure needed to complete a System Architecture. Use is made of the process guidance outlined in Figure 3.a, successive screens built into the ATS, a reusable database linked to all steps in the process, and a series of development tools that simplify and speed up the development process. All of this is built in a Microsoft Visual Studio framework, and in Visual J++ in Phase II, making future additions to the ATS a relatively easy process that does not require extensive software language skills. Any needed tool training will be provided by the PSI/MU team personnel, as part of a Phase 2 deliverable.

As stated previously, the ATS as proposed for the SBIR Phase II effort, will not be able to completely automate the process of synthesizing an OA into an SA. The main area of automation will be around the SA evaluation process, as this is the area that can be automated the most and provide the highest pay off. The following is a list of the areas that the ATS is anticipated to help the system architect:

- Message database feed into system architecture evaluation by providing an interface with the message database (also called the Information Exchange Requirements (IER) database). This is done by interfacing with either the C4RDP database (a rendition of the IER) or the Live Analyst Output (a tool that provides message loading through mission threads). Alternatively, the ATS will also interface with more loosely defined message databases when the detailed ones do not exist. This is done by: the automation of the feed of the Field Test data into a message database or by allowing the analyst to define a high level (i.e. not at the user/node level) canonical database.

- Functional architecture database feed into the system architecture evaluation – by providing an automated way to enter data not currently stored in the current system architecture's rendition. This data currently resides at the Management Information Bases (MIBs) used to initialize routers.

- Preliminary system architecture feed into the System Architecture evaluation by allowing the user to define a high level preliminary system architecture when the detailed one is not available.

- Systems Architecture Evaluation feed into the Systems Architecture Selection by providing visualization of the System Architecture results and providing automated input to the CECOM System Performance Models (SPM) and the Next Generation Performance Model (NGPM).

A 13-step process is shown in Table 3.a. The primary methods currently employed in evolving a system (primarily communication) from Step 2 (Operational Architecture) through Step 12 (Systems Architecture) involves highly time-consuming analytical and modeling approaches. Often poor designs that can be readily eliminated early in the process using high level analysis and modeling, are carried into later stages of the effort. This leads to greater expense and increased time use in the overall process.

13-STEP ARCHITECTURAL DEVELOPMENT PROCESS
1. Operational Requirements
2. Operational Architecture
3. Technical Requirements
4. System Specification
5. Preliminary System Architecture
6. Early Prototype Development
7. (a) System Architecture Evaluation
7. (b) C^2 Concept Evaluation
8. System Architecture Selection
9. Development Plans
10. Systems Implementation
11. Lab and Field Test and Evaluation
12. System Architecture

Table 3.a

We will briefly comment on each step listed in Table 3.a. *Operational requirements* are initially stated in advanced concept papers or formal requirements documents. The translation of these statements into an *operational architecture* is the result of work by skilled analysts and can involve an extensive effort. This involves developing different views of the OA as outlined in the AEA (see the list on Page 3, Appendix 1, of the Domain Analysis).

Technical requirements derived from operational requirements, involve the interaction of a system designer and the user who defined the operational requirements. Again, months of work may be involved. In the absence of any prototype systems, the user and developer are visualizing the system as it might eventually appear.
Once the technical requirements are written, a high-level *system specification* can be developed. Linkage to both the operational and technical requirements is essential. This, again, is a manpower-intensive effort.

The role of the systems' designer intensifies when a *preliminary system architecture* is formulated. This is an architecture that is characterized by a functional representation of the major components of the system. The components selected have the potential to meet performance and traffic requirements stated in the high-level specification. Measures of effectiveness must also be established. Several alternative functional representations can be created and a selection process (eliminating poor designs) can be initiated. If a model/emulation of the system/systems can be built at this time, this can be extremely useful. *Early prototype development* is often skipped in the current Army communication system development process. This step requires the use of special rapid prototyping tools that will be discussed in the next section. If a poor design can be eliminated at this point in the process, great cost

saving can be achieved. Steps 7(a) and 7(b) are aimed at *systems architecture* and C^2 *concept evaluation.* The first generally makes use of performance models (i.e., OPNET, GSS) and the latter involves man-in-the-loop simulation (HLA/DIS environment). Both are invaluable tools. However, the length of time and cost associated with building these models has proven to be a serious obstacle. Use of a hierarchical modeling approach (start with a high-level model and move to more detailed models only if needed) has long been employed in industry, while the Army has not always followed this approach.

Once evaluation has occurred, a selection process based on sound decision analysis is needed. *Systems architecture selection* should include a primary architecture recommendation and a series of less desirable but lower risk/cost alternatives as well. The less desirable alternatives can be used as part of a risk reduction strategy.

Development plans are essential in order to obtain needed budgets, staffing and scheduling of needed procurement contracts, exercises and tests. Most of the *systems implementation* by the RDEC and PEO's are done under contracts with industry and academia.

Laboratory and field tests are essential. They permit the system designer and user to assess how well the architecture has met the technical and operational requirements. With the use of test beds, battle labs and the Digital Integrated Laboratory, extensive user feedback is possible so that the eventual system fielded has a higher likelihood of success. However, many system faults are not discovered until this final phase of the effort. Corrections at this time are very expensive and often implemented as "band-aid" fixes.

Figure 3.a illustrates the enhanced development process. The same 13 steps shown in Table 3.a are illustrated along with a set of needed databases required to more effectively support the process. Building the database, its architecture, and being able to access the database in all phases of the development process will be one of the major objectives of the ATS development process. Also shown in the figure are the tools and processes needed to move from one step to the next.

4 Initial Specification

As stated previously, the ATS as it is being proposed for the SBIR Phase 2 effort, will not be able to completely automate the process of synthesizing an OA into an SA. The main area of automation will be around the SA evaluation process, as this is the area that can be automated the most and provide the highest pay off. The following is a list of the areas that the ATS is anticipated to help the system architect:

- Message database feed into system architecture evaluation by providing an interface with the message database (also called the Information Exchange Requirements (IER) database). This is done by interfacing with either the C4RDP database (a rendition of the IER) or the Live Analyst Output (a tool that provides message loading through mission threads). Alternatively, the ATS will also interface with more loosely defined message databases when the detailed ones do not exist. This is done by: the automation of the feed of the Field Test data into a message database or by allowing the analyst to define a high level (i.e. not at the user/node level) canonical database.

- Functional architecture database feed into the system architecture evaluation – by providing an automated way to enter data not currently stored in the current system architecture's rendition. This data currently resides at the Management Information Bases (MIBs) used to initialize routers.

- Preliminary system architecture feed into the System Architecture evaluation by allowing the user to define a high level preliminary system architecture when the detailed one is not available.

* Systems Architecture Evaluation feed into the Systems Architecture Selection by providing visualization of the System Architecture results and providing automated input to the CECOM System Performance Models (SPM) and the Next Generation Performance Model (NGPM).

A list of the detailed ATS requirements is included in the Phase II Work Plan Section of this proposal.

4.1 Model Specification

The models being proposed for the ATS SBIR Phase II effort are high-level (i.e., low fidelity level models) when compared to those currently used at CECOM for detailed System Performance Analysis (e.g., the System Performance Model (SPM) and the Next Generation Performance Model (NGPM)). The objectives of building high level models, even when high-resolution models exist, are the following:

* In the low-resolution models, the simulation running times are significantly smaller than those associated with the high-level resolution models. Because of this, one can afford to run a larger number of scenarios. The intent is to use the low-resolution models to filter out and find the most promising solutions, which can further examined using the high-resolution models. The low-resolution, in effect, allow us to sample the solution space.
* The low-resolution models will be a lot more flexible to modify, so that they can be used to analyze variations of a given system in a more expeditious manner. For example, if a new type of radio is being deployed and one desires to analyze the impact of deploying the Joint Tactical Radio System (JTRS) instead of the current data radio (i.e., the Enhanced Position Location Reporting System (EPLRS)). This can be more easily accommodated in a low-resolution model because the changes are not as extensive as they would be in the high-resolution model.

A list of the models identified in the Initial Specification is included below. This is the list of models that we then proceed to describe in the next sections.

* Single Channel Ground & Airborne Radio System (SINCGARS) and the MIL-STD-188-220 net access protocols.
* Enhanced Position Location Reporting System (EPLRS) to include the following capabilities: Switched Virtual Circuits (SVCs), MultiSource Group (MSG) needlines, and Carrier Sense Multiple Access (CSMAs) needlines).
* Near Term Data Radio (NTDR) – backbone only
* Internet Controller (INC) routing protocols (i.e., Internet Group Management Protocol (IGMP), Request For Comments (RFC) –1256, and Address Resolution Protocols (ARP)).
* Commercial Internet Routing Protocols (e.g., Open Shortest Path First(OSPF), Border Gateway Protocol(BGP), and Protocol Independent Multicast (PIM)).
* Situation Awareness
* Mobile Subscriber Equipment (MSE) to include the following capabilities: Circuit Switch, Packet Switch, and Asynchronous Transfer Mode (ATM).
* Network Manager and associated protocols (i.e., Simple Network Management Protocol (SNMP) and ISYSCON).
* Domain Name Server Models (DNS and TNS).
* Transport Protocols (TCP, UDP and NETBLT).
* Client-Server Applications (e.g., Remote Procedure Calls, Dynamic Host Control Protocol (DHCP)).
* Satellite Models (e.g., DAMA).
* Next Generation Protocols (e.g., Ipv6, Mobile IP, and mobile ad-hoc routing protocols).

4.1.1 Single Channel Ground & Airborne Radio System (SINCGARS) and the MIL-STD-188-220 net access protocols Models

The ATS will contain a SIP (System Improvement Product) SINCGARS and MIL-STD-188-220 model that can be invoked by the user. The low-resolution model will contain the following functionality:

- SINCGARS SIP Radio Functionality - The CSMA-like protocol used by the SIP SINCGARS radio to access the net. This includes the flywheel access algorithm implemented in the SIP radios that provides priority access to voice users and higher precedence users, as currently used by the Radio Embedded Net Access Delay (RE-NAD) in the Battlefield Digitization. Provisions to include the Deterministic Adaptive Priority Net Access Delay (DAP-NAD) used by the Fire Support Community will also be included. The model also provides for the "vulnerability" window that the SIP radios require to detect net busy conditions. The slot scheme and vulnerability window will be easily re-programmable to allow for expected improvements in the SIP radio. In addition, to provide for connectivity degradations, the user will be allowed to provide overall link qualities for each of the nets simulated as a function of time. This capability will allow the user to account for "stress" conditions in the network (e.g., jamming, fading, movement, and overall connectivity losses). The user will also be able to select effective transmission rates and information rates that allow the analyst to reflect potential improvements to the product (e.g., the Simultaneous Independent Voice and Data (SIVD) that allows Independent Data Transmissions while the net is occupied with voice).
- MIL-STD-188-220 Model Functionality – This model will contain the scheduler associated with the RE-NAD protocol that allows for staggering requests coming from the INC for net accesses (i.e., the INC will not request net access to the radio upon receiving data, but it waits for a scheduler "tick"). The scheduler timing depends on several factors: (1) the state of the queues of the stations in the network, (2) the highest precedence message at each of the stations in the network, and (3) voice utilization in the network. The factors in the scheduler should be easily re-programmable to accommodate for future changes in this area. In addition, this model will suspend timers when voice is present in the network (i.e., retransmission timers and scheduler timers). Type 1, Type 2, and Type 4 Layer frames will be modeled. This model also includes retransmissions and the transmission of acknowledgments. The different queues of the INC at the different precedence levels will be modeled (i.e., Net Control, Type 1, Type 4, Situation Awareness (SA) – UP and SA – DOWN queues). The Frame Multiple Unit (FMU) rules of the INC that determines how a transmission packet should be constructed (i.e., the order in which the INC queues will be serviced) will be followed. The order of queue servicing should be easily re-programmable to accommodate different alternatives.

4.1.2 Enhanced Position Location Reporting System (EPLRS) Model

The ATS will contain an EPLRS model that can be invoked by the user. The low-resolution model will contain the following functionality:

- EPLRS Switched Virtual Circuit (SVC) Radio Functionality - The EPLRS network offers a SVC service that is used by the Tactical Internet to exchange unicast type of data between INCs. The SVC capability of EPLRS requires that a dedicated circuit (also called needline) be setup at a specific rate (a rate that is specified by the user). After a timeout period, following last measured activity from the user, the INC tears down the needline so that the EPLRS resources can be used to setup other circuit demands. The ATS model will track timeslot resources that are allocated to setup needlines and will allow the setup and tear down of circuits. In addition, it will provide equivalent circuit paths as those expected in the EPLRS system.

- EPLRS CSMA needlines - The EPLRS Carrier Sense Multiple Access (CSMA) Needlines are used in the tactical internet for the distribution of the local Batttalion Situation Awareness and the distribution of Multicast addressed information. It has two different access schemes: CSMA-Normal (for messages longer than 240 bits) and CSMA-Short (for messages shorter or equal to 240 bits). It also has two different areas of coverage: CSMA-Local (with two levels of Relay) or CSMA-Extended (with 4 levels of relay). In addition, it includes an algorithm that reduces mutual interference in its relay scheme. The ATS EPLRS-CSMA model will include all of the modes of operation of the EPLRS CSMA (with their corresponding net access algorithms) and will include its mutual-interference reduction algorithm.
- EPLRS Multi-Source Group (MSG) - The EPLRS MSG needline is used in the Tactical Internet for the distribution of Situation Awareness (SA) between Battalions. It is also used for the injection of the Air Picture, the Enemy Picture, and the Position Location information derived by the Net Control Station (NCS) of the EPLRS system. The MSG needline allows up to 16 active sources and 120 radio relays. It divides the circuit into shares that are preallocated. This needline allows for automatically detecting the level of relay, which may be different for each source. The ATS EPLRS-MSG model will model the share splitting and the relay-level determination algorithm.

4.1.3 Near Term Data Radio (NTDR) Model

The ATS will contain an NTDR model that can be invoked by the user. The low-resolution model will contain the following functionality:

- The Media Access Control algorithms embedded into the NTDR system - The NTDR uses a Ready to Send (RTS)/ Clear to Send (CTS) mechanism to determine net access by a node. This mechanism allows for the destination node to send a signal to the source indicating that is ready to receive data. In addition, this information is also used by other neighboring nodes to stop their intended transmissions in order to avoid collisions. Transmission of the RTS is done randomly. In addition, transmissions are limited to 4 programmable step message sizes. The ATS NTDR Model will contain the net access mechanism of the NTDR and the step message sizing.
- NTDR Backbone Formation - The NTDR System allows for the formation of backbone and cluster networks. However, it is anticipated that the use of the NTDR is for intercommunication of Tactical Operation Centers (TOCs). When use for TOC-to-TOC communications, the NTDR will only be used with its backbone capability. As a result, modeling of the clustering algorithms is not considered necessary at this time. However, should it become a need, the ATS NTDR-model will be enhanced to provide for clustering formation. At this time, the ATS NTDR model will only contain the algorithm used by NTDR to determine the paths for communication between backbone nodes.

4.1.4 Internet Controller (INC) routing protocols Models

The INC routing protocol models will be: the Internet Group Management Protocol (IGMP), the Request For Comments (RFC) –1256 protocol, and the Address Resolution Protocols (ARP). A description of the functionality of these protocols is included in the following sections.

ATS IGMP Model Functionality

The IGMP Model Functionality will contain, at a minimum, the following properties:

- The multicast routing paths are equivalent to those expected in the actual system under similar scenario conditions
- The overhead associated with the implementation of the Multicast Routing is similar to that as expected in the actual system

- Its interface with the Upper Echelon Routing produces the same routes as expected in the actual system.

RFC-1256 Model Functionality

The ATS RFC-1256 Model Functionality will contain, at a minimum, the following properties:

- The unicast routing paths are equivalent to those expected in the actual system under similar scenario conditions
- The overhead associated with the implementation of the RFC-1256 Gateway selection is similar to that as expected in the actual system
- Its interface with the Upper Echelon Routing produces the same routes as expected in the actual system.

ARP Model Functionality

The ATS ARP Model Functionality will contain, at a minimum, the following properties:

- The unicast routing paths resulting from ARP requests are equivalent to those expected in the actual system under similar scenario conditions
- The overhead associated with the implementation of the ARP request and response is similar to that as expected in the actual system
- Its interface with the Upper Echelon Routing produces the same routes as those expected in the actual system.

4.1.5 *Commercial Internet Routing Protocols Models*

The Commercial Internet routing protocol models will be: the Open Shortest Path First (OSPF), the Border Gateway Protocol (BGP), and the Protocol Independent Multicast (PIM) – Sparse and Dense Version. The First Digitized Division has 2 Routing architecture alternatives to evaluate, according to the current thinking within the PEO C3S IPTs. The ATS models proposed in here will be of sufficient detail so that the routing alternatives can be analyzed.

Open Shortest Path First (OSPF) Model

The ATS OSPF Model Functionality will contain, at a minimum, the following properties:

- The unicast routing paths resulting from OSPF are equivalent to those expected in the actual system under similar scenario steady-state conditions (i.e., initialization and transient conditions will not be simulated).
- The overhead associated with system OSPF execution will be similar to that found in the model (i.e., for steady state conditions).
- Its interface with the Lower Echelon Routing produces the same routes as those expected in the actual system.

Border Gateway Protocol (BGP) Model

The ATS BGP Model Functionality will contain, at a minimum, the following properties:

- The unicast routing paths resulting from BGP are equivalent to those expected in the actual system under similar scenario steady-state conditions (i.e., initialization and transient conditions will not be simulated).
- The overhead associated with system BGP execution will be similar to that found in the model (i.e., for steady state conditions).
- Its interface with the Lower Echelon Routing produces the same routes as those expected in the actual system.

Protocol Independent Multicast (PIM) Model

The ATS PIM Model Functionality will contain, at a minimum, the following properties:

- The multicast routing paths resulting from PIM (dense and sparse mode) are equivalent to those expected in the actual system under similar scenario steady-state conditions (i.e., initialization and transient conditions will not be simulated).
- The overhead associated with system PIM execution will be similar to that found in the model (i.e., for steady state conditions).
- Its interface with the Lower Echelon Routing produces the same routes as those expected in the actual system.

4.1.6 Situation Awareness Model

The Situation Awareness model developed will be of sufficient detail that it will approximately provide the same SA load as that expected in the real system. In addition, it will take into account the dynamic SA coordination process .

4.1.7 Mobile Subscriber Equipment (MSE) Model

The ATS will contain a Mobile Subscriber Equipment model that can be invoked by the user. The low-resolution model will contain the following functionality:

- The Circuit Switching Portion of MSE – to include the trunk allocation algorithm and the Flood Search Routing conducted by MSE to find a path between source and destination subscribers.
- The Packet Switching Portion of MSE – to include the breakdown of messages into packets routed by Shortest Path First Routing algorithm
- The Asynchronous Transfer Mode (ATM) Portion of MSE – to include the various services offered by ATM (i.e., the Constant Bit Rate Service, the Variable Bit Rate Service, and the Available Bit Rate Service).

4.1.8 ATS Network Manager Model

The ATS will contain a Network Manager model that can be invoked by the user. This model is of low resolution and will only simulate expected overhead due to network management activities during steady state conditions performed via Simple Network Management Protocol (SNMP) and the Integrated System Control Center (ISYSCON).

4.1.9 Domain Name Server Models

The ATS will contain a Domain Name Server model that can be invoked by the user. This model is of low resolution and will only simulate expected overhead due to Domain Name Server and Tactical Name Server during steady state conditions.

4.1.10 Transport Protocols (TCP, UDP and NETBLT).

The ATS will contain a Transport Protocol Model that can be invoked by the user. These models are of low resolution and will only simulate expected overhead due to retransmissions and acknowledgments. The protocols included in the model are: Transmission Control Protocol (TCP), User Datagram Protocol (UDP), and Net Block Transfer (NETBLT).

4.1.11 ATS Client-Server Applications (e.g., Remote Procedure Calls(RPC), Dynamic Host Control Protocol (DHCP))

The ATS will contain Client Server Application Models. These models are of low resolution and will only simulate expected overhead during steady state conditions.

4.1.12 Satellite Models (e.g., DAMA).

The ATS will contain Client Server Application Models. These models are of low resolution and will only simulate expected overhead and long delays during steady state conditions

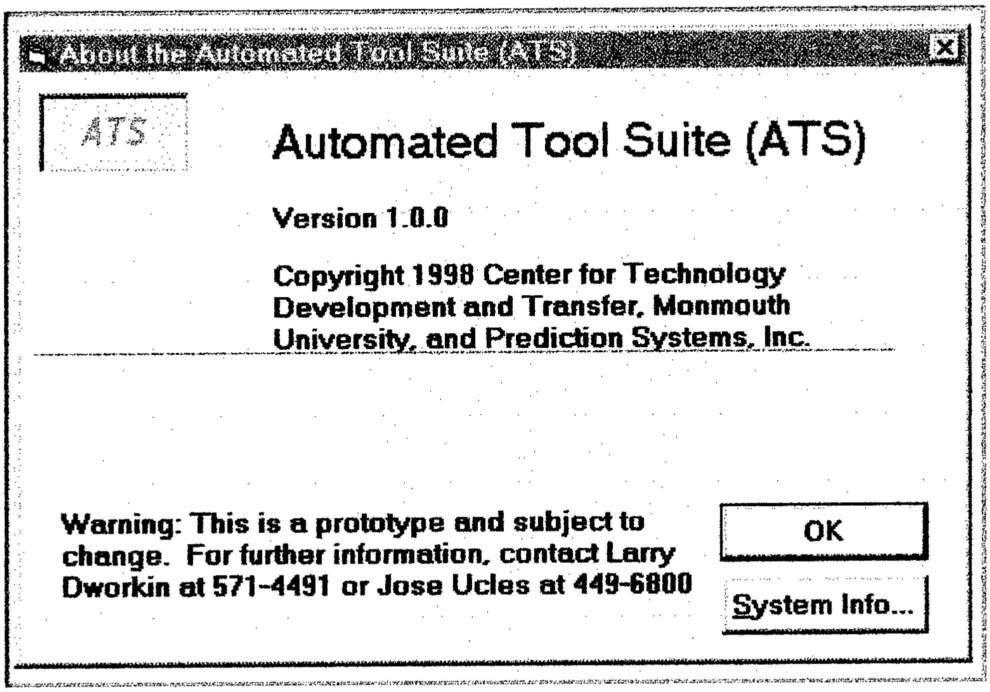
4.1.13 Next Generation Protocols (e.g., Ipv6, Mobile IP, and mobile ad-hoc routing protocols).

The ATS will be easily modifiable to include Next Generation Protocols such as: Internet Protocol – Version 6, Mobile IP, and Ad-Hoc Routing Protocols such as the ones proposed in the Global Mobile (GloMo) DARPA program.

5 Software Specification Report

The software specification report is a description of the screens developed for the ATS prototype using Microsoft Visual Studio. These ATS screens in conjunction with other tools and databases will lead the user, in a logical fashion through the process of creating a systems architecture from an operational architecture. The screens were documented in the Concept of Operation Report and will not be repeated in here.

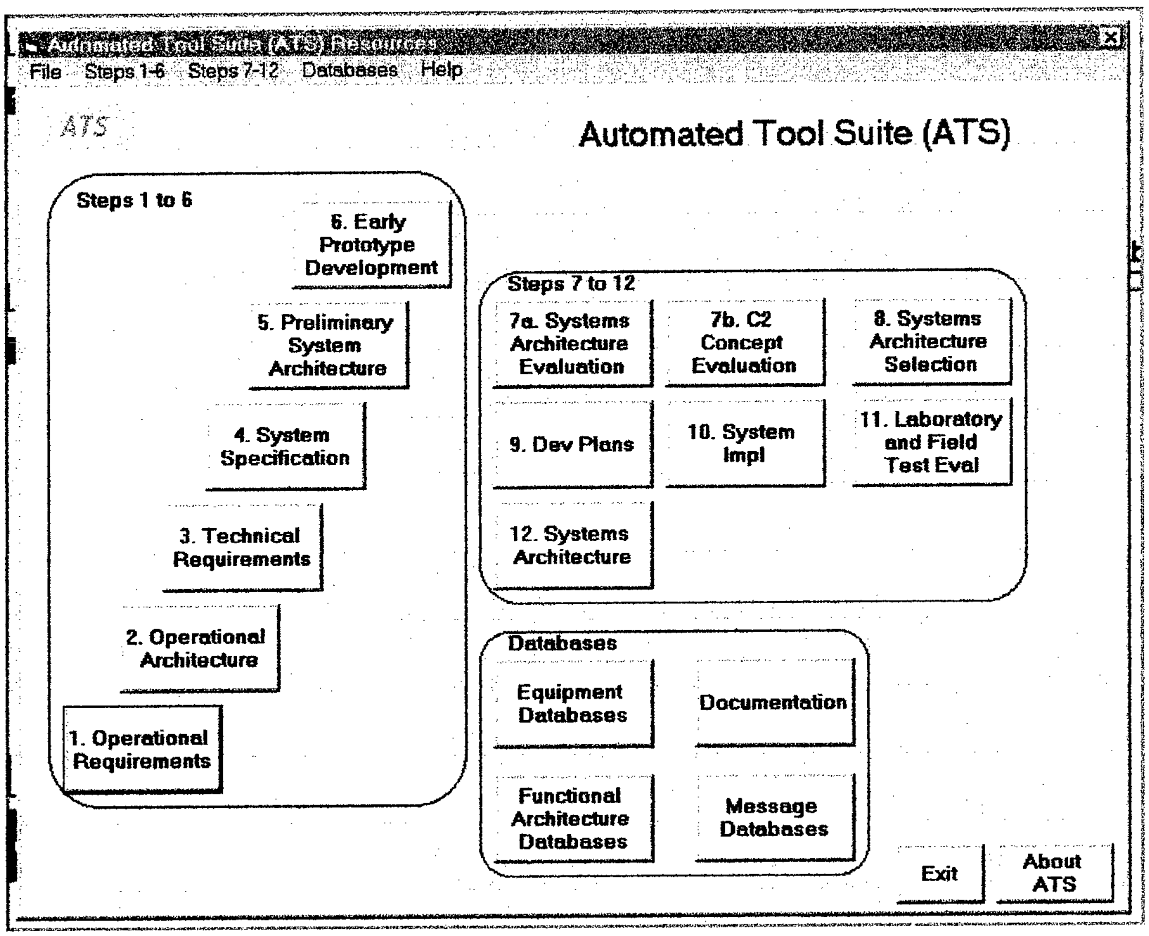
The following documents each of the screens available in the ATS prototype, except for Screen 1 that contains the title of the project (i.e., Automated Tool Suite (ATS)), the version number (i.e., 1), copyright statement, and points of contact.



Upon clicking on the OK box, the screen shown in the next Figure appears. The user has the option of starting a new architecture development or embellishing an existing systems architecture. In either case, he/she advances to the screen shown in the next Figure.

Shown are the 13 steps and supporting databases along with necessary control buttons (e.g., Exit, Help).

Architecture Development Options Screen



Operational Architecture Screen

Name Operational Architecture

Explanation An operational architecture development process results in the allocation of operational, organizational, and physical elements to the activities necessary to execute the operation concept.

Next Step Button
- Message Data Base
- Technical Requirements
- Equipment Data Base

Work in Window Buttons
- Live Analyst
- Netviz
- Documentation
- Operational Concept Description – FM 100-5

Technical Requirements Screen

Name Technical Requirement

Explanation The technical requirements are the basis for the technical design. The conversion of verbal descriptions of these requirements will be refined through rapid prototyping and early user feedback.

Next Step Button
- CAPS
- Object time

Work in Window Buttons
- Documentation
- Requirement Specification Edit
- Operation Requirement Matrix Cross Reference (new)

System Specification Screen

Name System Specification

Explanation The System Specification defines the Operational Architecture Interface, Systems Architecture Interfaces, Generic System Inputs, Output Interfaces and System Performance Characteristics.

Next Step Buttons
- Preliminary System Architecture
- System Development Plans (Step 9)

Work in Window Buttons
- Documentation
- Product Specification
- Technical Requirement Cross Reference Matrix (new)

Preliminary System Architecture Screen

Name — Preliminary System Architecture

Explanation — The preliminary system architecture is a high level view of a system architecture. Architecture functional block diagrams, general description of services and traffic supported, identifying quality of service, general employed practical description and other key characteristics such as desired performance, mobility, interoperability, survivability, and cost and risk should be identified.

Next Step Buttons

- Technical Requirement Beedback (Step 3)
- Decision Analysis Tool
- Early Prototype Development (Step 6)

Work in Windows Buttons

- Excel Spreadsheet
- Architecture Drawing Tool
- Model Selection Tool (Knowledge Based)
- High Level Modeling Tool Selection
- Analysis Support Tool

Early Prototype Development Screen

Name — Early Prototype Development

Explanation — Through the use of rapid prototyping tools (e.g., CAPS, Objective) a high level early prototype of the communication/processing system architecture will be created. This early prototype will involve iterative evaluation on the part of a user or permit high level performance evaluation on the part of the designer.

Next Step Buttons

- Technical Requirement Feedback
- C2 Concept Evaluation
- Systems Architecture Evaluations
- Documentation

Work in Box Buttons

- CAPS
- Object Time
- Microsoft Visual Studio Utilities
- High level Modeling Tool Selection

System Architecture Evaluation Screen

Name — Systems Architecture Evaluation

Explanation — Through the use of analysis and simulation, alternative design for communication/processing systems will be conducted. Critical performance characteristics will be determined. Necessary factual information for architecture prioritization and selection will be created and documented.

Next Step Buttons

- Decision Analysis Tool
- Excel Spreadsheet
- Documentation

Work in Box Buttons

- Message Data Base
- Equipment Data Base
- Functional Architecture Data Base
- Systems Architecture Selection
- Performance Model

C2 Concept Evaluation Screen

Name C2 Concept Evaluation

Explanation Through the use of HLA-compliant man-in-the-loop simulation environments, the C2 capability of system will be assessed. Use of early prototypes, existing HLA-compliant simulations and performance modeling is involved.

Next Step Buttons

- Operational Requirement (Step 1) Feedback
- Development Plans (Step 9)
- System Implementation (Step 10)
- System Architecture Selection (Step 8)
- Early Prototype Development Feedback (Step 8)

Work in Box Buttons

- HLA-Compliant Models/Interfaces
- External Scenarios Drivers (constructive models)
- Message Data Base
- Documentation

System Architecture Selection Screen

Name Systems Architecture Selection Explanation

Explanation The use of decision analysis techniques will be applied to the results obtained from performance analysis/tools and C2 concept testing man-in-the-loop simulations. Architecture alternatives will be prioritized and subject to sensitivity analysis.

Next Step Buttons

- Development Plans (Step 9)
- C2 concept Evaluation (Step 7b)
- Systems Architecture Evaluation (Step 7a)

Work in Box Buttons

- Expert Choice Evaluation Tool
- Excel Spreadsheet
- Microsoft Money
- Documentation

Development Plans Screen

Name Development Plans

Explanation The creation of a development plan for the building, evaluation and testing of the system architecture is the primary function of this step. This involves project planning, scheduling, costing and presentation preparation.

Next Step Buttons

- System Implementation (Step 10)

Work in Box Buttons

- Microsoft Office
- Microsoft Project
- Microsoft Money
- Documentation
- Procurement Package

System Implementation Screen

Name System Implementation

Explanation Implementation of the System Architecture by contractor or in-house personnel.

Next Step Button

- Laboratory and Field Test Evaluation (Step 11)
- C2 Concept Evaluation (Step 7b)
- Development Plans (Step 9)

Work In Box Buttons

- Documentation
- Test Plans
- Microsoft Project
- HLA man-in-the-loop simulation
- OPNET model connection
- Functional Architecture Data Base
- Microsoft Budgeting Tool (?)

Laboratory and Field Test Evaluation Screen

Name Laboratory and Field Test Evaluation

Explanation Support required to conduct laboratory and field test.

Next Step Buttons

- Technical Requirement Feedback (Step 3)
- Equipment Data Base
- Message Data Base
- System Architecture Field Data Collection (into documentation)

Work in Box Buttons

- Input Data From DIL (Message DB)
- Field Test Results (Message DB)
- Documented Test Plans (Others)

6 Software Report

The Software report contains a copy of the source code (this is not available in the electronic version of this report – only the hard copy) and is included in Appendix A that follows.

Appendix A

```
Option Explicit

' Reg Key Security Options...
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE + _
                       KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS + _
                       KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                         ' Unicode nul terminated string
Const REG_DWORD = 4                      ' 32-bit number

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias "RegOpenKeyExA" (ByVal hKey As Long,
ByVal lpSubKey As String, ByVal ulOptions As Long, ByVal samDesired As Long, ByRef phkResult As
Long) As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias "RegQueryValueExA" (ByVal hKey As
Long, ByVal lpValueName As String, ByVal lpReserved As Long, ByRef lpType As Long, ByVal lpData
As String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As Long) As Long


Private Sub cmdSysInfo_Click()
  Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
  Unload Me
End Sub

Private Sub Form_Load()
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
End Sub

Public Sub StartSysInfo()
    On Error GoTo SysInfoErr

    Dim rc As Long
    Dim SysInfoPath As String

    ' Try To Get System Info Program Path\Name From Registry...
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO, gREGVALSYSINFO, SysInfoPath) Then
    ' Try To Get System Info Program Path Only From Registry...
    ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC, gREGVALSYSINFOLOC, SysInfoPath) Th
en
        ' Validate Existance Of Known 32 Bit File Version
        If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
            SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

        ' Error - File Can Not Be Found...
        Else
            GoTo SysInfoErr
        End If
    ' Error - Registry Entry Can Not Be Found...
    Else
        GoTo SysInfoErr
    End If
```

```
    Call Shell(SysInfoPath, vbNormalFocus)

    Exit Sub
SysInfoErr:
    MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub

Public Function GetKeyValue(KeyRoot As Long, KeyName As String, SubKeyRef As String, ByRef KeyVa
l As String) As Boolean
    Dim i As Long                                           ' Loop Counter
    Dim rc As Long                                          ' Return Code
    Dim hKey As Long                                        ' Handle To An Open Registry Key
    Dim hDepth As Long                                      '
    Dim KeyValType As Long                                  ' Data Type Of A Registry Key
    Dim tmpVal As String                                    ' Tempory Storage For A Registry Key
 Value
    Dim KeyValSize As Long                                  ' Size Of Registry Key Variable
    '------------------------------------------------------------
    ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
    '------------------------------------------------------------
    rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) ' Open Registry Key

    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle Error...

    tmpVal = String$(1024, 0)                             ' Allocate Variable Space
    KeyValSize = 1024                                       ' Mark Variable Size

    '------------------------------------------------------------
    ' Retrieve Registry Key Value...
    '------------------------------------------------------------
    rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
                         KeyValType, tmpVal, KeyValSize)    ' Get/Create Key Value

    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle Errors

    If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then           ' Win95 Adds Null Terminated String.
..
        tmpVal = Left(tmpVal, KeyValSize - 1)               ' Null Found, Extract From String
    Else                                                    ' WinNT Does NOT Null Terminate Stri
ng...
        tmpVal = Left(tmpVal, KeyValSize)                   ' Null Not Found, Extract String Onl
y
    End If
    '------------------------------------------------------------
    ' Determine Key Value Type For Conversion...
    '------------------------------------------------------------
    Select Case KeyValType                                  ' Search Data Types...
    Case REG_SZ                                             ' String Registry Key Data Type
        KeyVal = tmpVal                                     ' Copy String Value
    Case REG_DWORD                                          ' Double Word Registry Key Data Type
        For i = Len(tmpVal) To 1 Step -1                    ' Convert Each Bit
            KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1)))   ' Build Value Char. By Char.
        Next
        KeyVal = Format$("&h" + KeyVal)                     ' Convert Double Word To String
    End Select

    GetKeyValue = True                                      ' Return Success
    rc = RegCloseKey(hKey)                                  ' Close Registry Key
    Exit Function                                           ' Exit

GetKeyError:    ' Cleanup After An Error Has Occured...
    KeyVal = ""                                             ' Set Return Val To Empty String
    GetKeyValue = False                                     ' Return Failure
    rc = RegCloseKey(hKey)                                  ' Close Registry Key
End Function
```

```
Private Sub about_Click()
frmAbout.Show modal
End Sub

Private Sub cmdAbout_Click()
frmAbout.Show modal
End Sub

Private Sub cmdDone_Click()
Unload Me
End Sub


Private Sub Option4_Click()
MsgBox "Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub


Private Sub cmdEdit_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub com_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub cpm_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub dd_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub ddd_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub exit_Click()
Unload Me

End Sub

Private Sub fsm_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub idd_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub irs_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub
```

```
Private Sub ocd_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\winword.exe 498ocd.doc", vbMaximizedFoc
us)
End Sub

Private Sub scom_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub sdd_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub sdp_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub siom_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub sip_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub sps_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub srs_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub sss_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub std_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub stp_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub

Private Sub str_Click()
```

```
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub strp_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub sum_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub svd_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub

Private Sub title_Click()
frmTitle.Show modal
End Sub

Private Sub um_Click()
MsgBox "ATS Document - Not Yet Implemented", 48, "ATS - Not Implemented"

End Sub
```

```
Private Sub cmdMyCaption_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\winword.exe", vbMaximizedFocus)
End Sub

Private Sub cmdFrank_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\winword.exe", vbMaximizedFocus)
End Sub

Private Sub cmdOCD_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\winword.exe 498ocd.doc", vbMaximizedFoc
us)
End Sub

Private Sub Command1_Click()

End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdNetviz_Click()
Result = Shell("C:\netViz 3.0 Demo\Program\NVDEMO.EXE", vbMaximizedFocus)
End Sub

Private Sub cmdOther_Click()

End Sub

Private Sub cmdTechReq_Click()
frmOp03.Show modal
Unload Me
End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdLiveAnalyst_Click()

End Sub

Private Sub cmdNetviz_Click()

End Sub

Private Sub cmdOCD_Click()

End Sub


Private Sub cmdOther_Click()

End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdOCD_Click()

End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdOCD_Click()

End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdProject_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\WINPROJ.EXE", vbMaximizedFocus)
End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub cmdProject_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\WINPROJ.EXE", vbMaximizedFocus)
End Sub

Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub mnuExit_Click()
Unload Me
End Sub

Private Sub mnuHelpAbout_Click()
frmAbout.Show modal
End Sub

Private Sub mnuHelpST_Click()
frmTitle.Show modal
End Sub
```

```
Private Sub Command16_Click()

End Sub

Private Sub about_Click()
frmAbout.Show modal
End Sub

Private Sub archsel_Click()
frmOp08.Show modal
End Sub

Private Sub banner_Click()
frmTitle.Show modal
End Sub

Private Sub c2concept_Click()
frmOp07b.Show modal
End Sub

Private Sub cmdAbout_Click()
frmAbout.Show modal
End Sub

Private Sub cmdDBDocumentation_Click()
frmDocumentation.Show modal
End Sub

Private Sub cmdDBEquipment_Click()
MsgBox "This Button is not yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub cmdDBFunctional_Click()
MsgBox "This Button is not yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub cmdDBMessage_Click()
Result = Shell("C:\Program Files\Microsoft Office\Office\EXCEL.EXE Figure4.xls", vbMaximizedFocu
s)

End Sub

Private Sub cmdDone_Click()
Unload Me
End Sub


Private Sub cmdEvalConcept_Click()
frmOp07b.Show modal

End Sub

Private Sub cmdEvalLab_Click()
frmOp11.Show modal

End Sub


Private Sub cmdEvalPrototype_Click()
frmOp06.Show modal

End Sub
```

```
Private Sub cmdReqOperational_Click()
frmOp01.Show modal
End Sub

Private Sub cmdReqTechnical_Click()
frmOp03.Show modal

End Sub


Private Sub cmdSpecOpArch_Click()
frmOp02.Show modal

End Sub


Private Sub cmdSpecSystem_Click()
frmOp04.Show modal

End Sub


Private Sub cmdSysArchDefinition_Click()
MsgBox "This Button is not yet Implemented", 48, "ATS - Not Implemented"

End Sub


Private Sub cmdSysArchEvaluation_Click()
frmOp07a.Show modal

End Sub


Private Sub cmdSysArchPreliminary_Click()
frmOp05.Show modal

End Sub


Private Sub cmdSysArchSelection_Click()
frmOp08.Show modal

End Sub


Private Sub Command1_Click()
frmOp09.Show modal
End Sub


Private Sub cmdSysImplementation_Click()
frmOp10.Show modal

End Sub


Private Sub dbdoc_Click()
frmDocumentation.Show modal
End Sub

Private Sub devplans_Click()
frmOp09.Show modal
End Sub

Private Sub earlyproto_Click()
frmOp06.Show modal
End Sub
```

```
Private Sub exit_Click()
Unload Me
End Sub

Private Sub Form_Load()
frmOp01.Show modal
End Sub

Private Sub labeval_Click()
frmOp11.Show modal
End Sub

Private Sub oparch_Click()
frmOp02.Show modal
End Sub


Private Sub prelimsysarch_Click()
frmOp05.Show modal
End Sub

Private Sub sysarch_Click()
frmOp12.Show modal
End Sub

Private Sub sysarcheval_Click()
frmOp07a.Show modal
End Sub

Private Sub sysimp_Click()
frmOp10.Show modal
End Sub

Private Sub sysspec_Click()
frmOp04.Show modal
End Sub

Private Sub techreq_Click()
frmOp03.Show modal
End Sub
```

```
Option Explicit

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
End Sub

Private Sub Form_Load()
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblProductName.Caption = App.Title
End Sub

Private Sub Frame1_Click()
    Unload Me
End Sub
```

```
Private Sub about_Click()
frmAbout.Show modal

End Sub

Private Sub cmdAbout_Click()
frmAbout.Show modal

End Sub


Private Sub cmdDone_Click()
Unload Me

End Sub

Private Sub cmdNew_Click()
frmResources.Show modal
End Sub


Private Sub cmdOld_Click()
frmResources.Show modal
End Sub


Private Sub create_Click()
frmResources.Show modal

End Sub


Private Sub edit_Click()
frmResources.Show modal

End Sub


Private Sub exit_Click()
Unload Me

End Sub


Private Sub title_Click()
frmTitle.Show modal
End Sub


Private Sub um_Click()
MsgBox "ATS - Not Yet Implemented", 48, "ATS - Not Implemented"
End Sub
```

```
Option Explicit

Private Sub Command1_Click()
frmStartup.Show
Unload Me
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
    Unload Me
End Sub

Private Sub Form_Load()
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor & "." & App.Revision
    lblProductName.Caption = App.Title
End Sub

Private Sub Frame1_Click()
    Unload Me
End Sub
```

References

1. *Department of the Army Enterprise Architecture Framework*, Version 1.0X, 15 Aug 1997.
2. IDEF.0 and IDEF.1X for a C^4I Army Division.
3. OPNET Users Manual.
4. GSS Users Manual.
5. Reports on AMASE submitted under Contrtact DAAH04-93-G0401, 1994-97.
6. Netviz User Manual.
7. Live Analyst User Manual.
8. Web information at URL—http://www.sei.cmu.edu/technology/architecture/sites.html (on software architectures)
9. TIMS—SRI Presentation, L. David Boschert.
10. *System Architecture Tool Design*, PSI Proposal on BAA, Topic #97-08.
11. *Application of Feature-Oriented Domain Analysis to the Army Movement Central Domain*, CMU-SEI, S. Cohen et al, June 1992.
12. *Modeling Guidelines for Object-Oriented Analysis*, Institute of Software Systems Techniques, Fraunhafer, G.E., August 1997.

Class Notes, *Domain Engineering*, Software Engineering Department, Monmouth University, Professor Richard Conn, September 1996.